A PARALLEL IMPLEMENTATION OF A TIME-DEPENDENT, TWO-DIMENSIONAL FLAME MODEL WITH DETAILED CHEMISTRY

G. Patnaik
Berkeley Research Associates

D. Fyfe and K. Kailasanath
Naval Research Laboratory

B. Moon, R. Bennett, A. Sussman and J. Saltz
University of Maryland

Introduction

Physical processes in hydrocarbon flames are highly complex and interact in a strongly non-linear fashion. Numerical experimentation is an excellent way to isolate physical processes, study their interactions, or predict important properties such as flammability limits. Only highly detailed models that include complex chemistry and diffusive processes can obtain the correct flammability limits. To date, sufficiently detailed calculations for hydrocarbon flames have only been carried out for steady-state flames. However, the extinction of hydrocarbon flames is a multidimensional, transient process. Numerous two-dimensional calculations of detailed hydrogen flames [1-3] and some preliminary calculations of transient methane flames with moderately detailed chemistry have been carried out at the Naval Research Laboratory [4]. For heavier hydrocarbon fuels, which are of more practical interest, the computational requirements are currently beyond the capabilities of current vector supercomputers. Thus it is imperative that a detailed flame code be ported to massively parallel computers, which have the potential to handle these problems.

The numerical model used in the NRL flame code [5] is predominantly based on structured finite volume methods. The chemistry is modeled by a system of ordinary differential equations which is solved independently at each grid point. Thus, though the model uses a mesh structure, the workload at each grid point can vary considerably. It is this feature that requires the use of both structured and unstructured methods in the same code.

Parallel Implementation

The solution methods for the various physical processes in the flame can be placed into one of two groups:

Structured: where the computation is based on structured meshes. Certain processes, e.g. heat conduction, are represented by partial differential equations which are discretized by a finite volume technique. In these methods, the solution procedure requires a good deal of communication with neighboring points, but the amount of computation at each point is nearly the same.

Independent: where computation at each grid point can be carried out independently without requiring any communication between neighboring points, e.g. radiation, chemistry. However, the solution procedure may require a different amount of computation at each grid point. This distribution of work load will change from time step to time step as the flame evolves.

The Multiblock PARTI runtime support library [6] has been developed to parallelize multiblock and multigrid codes on distributed memory (MIMD) machines. This library allows programmers to lay out distributed arrays in a flexible way, give high level description for performing data movement, and distribute computation across processors. Multiblock PARTI has been used in the NRL flame code for all structured processes. A multigrid method is used as the solution technique for the elliptic equation that arises in the fluid convection model. The restriction and prolongation operations for shifting between different multigrid levels require moving regular array sections with non-unit strides. These operations have been implemented using Multiblock PARTI.

The PARTI library was incorporated into the existing sequential flame code with after some alterations to the code. It is important that the data and temporary working array layout in the sequential code is suitable for domain decomposition, but usually this is not true in existing serial and vector programs. Changing the layout to make it suitable proved to be quite time consuming. The naive approach of promoting all temporaries to arrays is wasteful of memory and is not needed on MIMD machines since only data that is involved in near-neighbor operations needs to be promoted.

To balance the work load of the chemistry process, the block-partitioned data must be redistributed across processors, and then returned to their original locations for the next structured process. This approach inherently requires a substantial amount of communication at every time step. Two new algorithms, Binpack and Binsort, which perform this redistribution of work load and reduce communication volume at low cost have been implemented using the CHAOS library [7]. The Binpack algorithm determines which processors are overly loaded and which processors are lightly loaded, computes the amount of work that must be moved from overly loaded processors to lightly loaded processors, and then generates a load balancing plan. Binsort is a heuristic for improving the performance of Binpack by reducing the communication volume for redistributing the work load. The Binsort algorithm achieves this by selecting for movement those grid points with greater amounts of work; thus fewer number of grid points need to be considered by Binpack to attain a load balance. A good estimate of the work load is needed, however is hard to obtain this for a sophisticated ODE solver. Once an estimate is made however, Binpack will balance the load almost exactly.

Results

The parallel flame code has been implemented on the Intel iPSC/860 and Paragon computers. Calculations have been performed on an extinguishing 4.5% methane-air flame in a 5.1 cm isothermal channel. The OH mole fraction at 0.18 secs. is presented in Fig. 1. These preliminary results suggest that the extinguishment mechanism may be similar to that in a hydrogen-air flame [3].

For this methane computation on a 256 X 256 grid, it requires 50 nodes of the Intel Paragon to equal one CRAY C-90 processor. The timings of the important processes is shown in Fig. 2. All processes except the fluid convection scale well. The fluid convection does not scale due to the large amount of communication and scalar code required by the multigrid solution procedure.

The chemistry takes 60 - 70% of the total time; thus it is very important that a good load balance is achieved. The workload distribution (in arbitrary units) at a late time is shown in Fig. 3. The workload in the two spots near the wall is as high as 80 times the smallest workload. Load balancing improves the chemistry run time by 30% for an overall savings of 20%.

Conclusions

An extinguishing methane-air flame has been simulated using a detailed, two-dimensional. time-dependent. numerical model on the Intel iPSC/860 and Paragon parallel computers. The performance of a single processor CRAY C-90 can be equaled with 50 Paragon processors. Since Paragon systems with hundreds of processors are available, it is now possible to solve very large problems.

The PARTI software is available on various other MIMD computers, and the flame code will be implemented on these other computers as well, especially the IBM SP-2. The SP-2 is expected to be 5-10 times faster than the Intel Paragon. With the development of the parallel flame code on this machine , it will be possible to attempt three-dimensional, time-dependent calculations with detailed hydrocarbon chemistry for the first time.

Acknowledgements

This work was sponsored by NASA in the Microgravity Science and Applications Program and the Office of Naval Research through the Naval Research Laboratory.

References

1. Patnaik, G., Kailasanath, K., Laskey, K., and Oran, E.S., Detailed Numerical Simulations of Cellular Flames. In *Proceedings of the 22nd Symposium (Intl.) on Combustion,* pp. 1517–1526, Combustion Institute, 1988.
2. Patnaik, G. and Kailasanath, K., Effect of Gravity on the Stability and Structure of Lean Hydrogen – Air Flames, in *Twenty-third Symposium (Intl.) on Combustion,* pp. 1641-1647, Combustion Institute, 1991.
3. Patnaik, G. and Kailasanath, K., Numerical Simulations of the Extinguishment of Downward Propagating Flames, in *Twenty-fourth Symposium (Intl.) on Combustion,* pp. 189-195, Combustion Institute, 1992.
4. Patnaik, G. and Kailasanath, K., Cellular Structure of Lean Hydrogen and Methane Flames, AIAA Paper 93-3275, AIAA, 1994.
5. Patnaik, G., Laskey, K., Kailasanath, K., Oran, E.S. and Brun, T.A., FLIC — A Detailed, Two-Dimensional Flame Model, Memorandum Report 6555, Naval Research Laboratory, 1989.
6. Agrawal, G., Sussman, A. and Saltz, J., Compiler and Runtime Support for Structured and Block Structured Applications, in *Proceedings Supercomputing '93,* pp. 578-587, IEEE Computer Society Press, 1993.
7. Das, R., Uysal, M., Saltz, J. and Hwang, Y., Communication Optimizations for Irregular Scientific Computations on Distributed Memory Architectures, Department of Computer Science Report CS-TR-3163, University of Maryland, 1993.

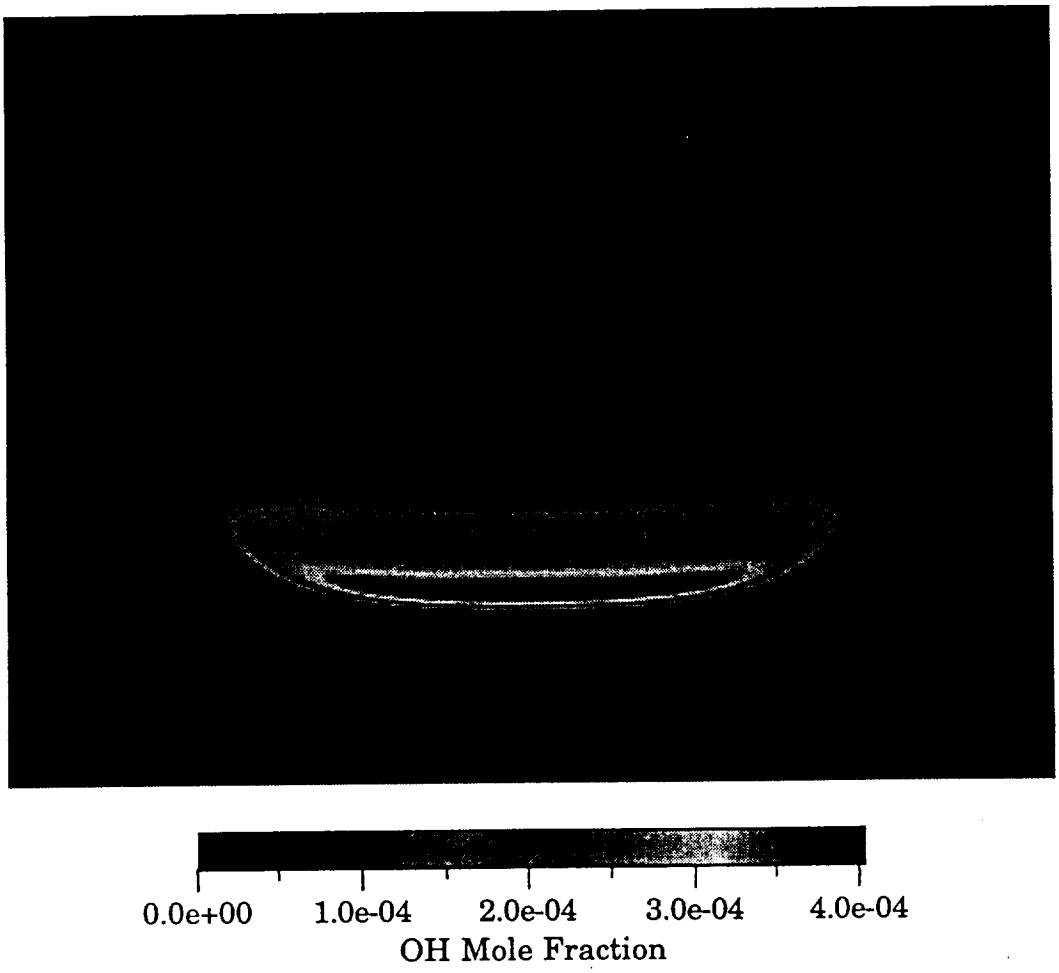

Figure 1. OH Mole Fraction at 0.18 secs.